ΨΗΦΙΑΚΕΣ ΤΗΛΕΠΙΚΟΙΝΩΝΙΕΣ

Ονοματεπώνυμο: Δούρου Βασιλική Ευαγγελία

Α.Μ.: 1072633

Έτος: 4ο

Ερωτήσεις-Ζητούμενα για Μέρος 1:

Η μέθοδος PCM είναι μία μέθοδος κωδικοποίησης κυματομορφής που μετατρέπει ένα αναλογικό σήμα σε ψηφιακά δεδομένα. Αποτελείται από έναν δειγματολήπτη, έναν κβαντιστή και έναν κωδικοποιητή.

Σε αυτή τη μέθοδο μπορεί να χρησιμοποιηθεί και ένας ομοιόμορφος και ένας μη ομοιόμορφος κβαντιστής. Στη περίπτωση του μη ομοιόμορφου κβαντιστή επιτρέπεται οι περιοχές κβάντισης να μην έχουν το ίδιο εύρος. Έτσι, ενώ λειτουργεί ικανοποιητικά για σήματα με ομοιόμορφη κατανομή εισόδου, έχει και καλύτερες επιδόσεις σε μη ομοιόμορφες πηγές, το οποίο τον κάνει να υπερτερεί σε σχέση με έναν ομοιόμορφο κβαντιστή.

Στα πλαίσια της άσκησης, για τη μη ομοιόμορφη κβάντιση θα χρησιμοποιηθεί ο αλγόριθμος Lloyd Max. Για τον αρχικό υπολογισμό των επιπέδων κβάντισης, επιλέγονται τα κέντρα ενός ομοιόμορφου κβαντιστή και στη συνέχεια ο αλγόριθμος εκτελεί K_{max} επαναλήψεις. Στη διάρκεια της κάθε επανάληψης i υπολογίζονται τα όρια των ζωνών κβάντισης, το κβαντισμένο σήμα, η παραμόρφωση D_i και τα νέα επίπεδα κβάντιστης.

Επιπλέον, στα πλαίσια του 1ου μέρους της άσκησης, ζητείται και η υλοποίηση της προσαρμοστικής διαμόρφωσης Δέλτα. Η ADM συγκεκριμένα, διατηρεί την απλότητα της DM, χρησιμοποιώντας όμως μεταβαλλόμενο βήμα. Για μικρή κλήση του σήματος εισόδου, το βήμα της ADM μικραίνει ώστε να αποφευχθεί ο κοκκώδης θόρυβος, ενώ όταν η κυματομορφή του σήματος εμφανίζει απότομη κλίση, αυξάνεται το βήμα της ADM.

1.1.α. Οι $AR_1(1)$ και $AR_2(1)$ διαδικασίες υλοποιήθηκαν σύμφωνα με τις οδηγίες της άσκησης, με $\alpha_1 = 0.9$ και $\alpha_1 = 0.01$ αντίστοιχα, και χρησιμοποιήθηκαν οι συναρτήσεις min και max της MatLab για τον υπολογισμό της ελάχιστης και της μέγιστης τιμής αντίστοιχα.

Έτσι, προέκυψε για την διαδικασία $AR_1(1)$ $min_{value} = -9.2054$ και $max_{value} = 8.6100$, ενώ για τη διαδικασία $AR_2(1)$ προέκυψαν $min_{value} = -3.7476$ και $max_{value} = 3.5818$.

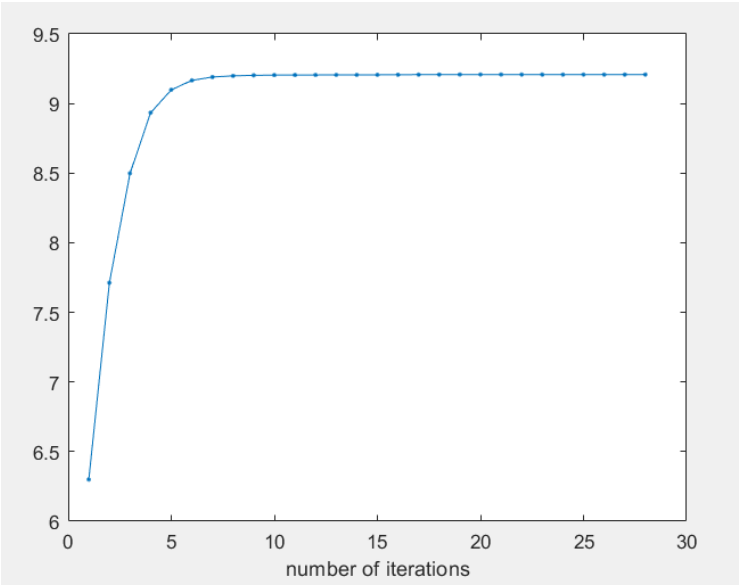
Οι τιμές του SQNR σε dBs που προέκυψαν μετά από K_{max} επαναλήψεις χρησιμοποιώντας PCM με N=2, 4 και 8 bits είναι οι ακόλουθες:

N	$SQNR\ AR_1(1)$	$SQNR\ AR_2(1)$
2	9.208132	9.306947
4	20.358685	20.138268
8	41.432244	41.840560

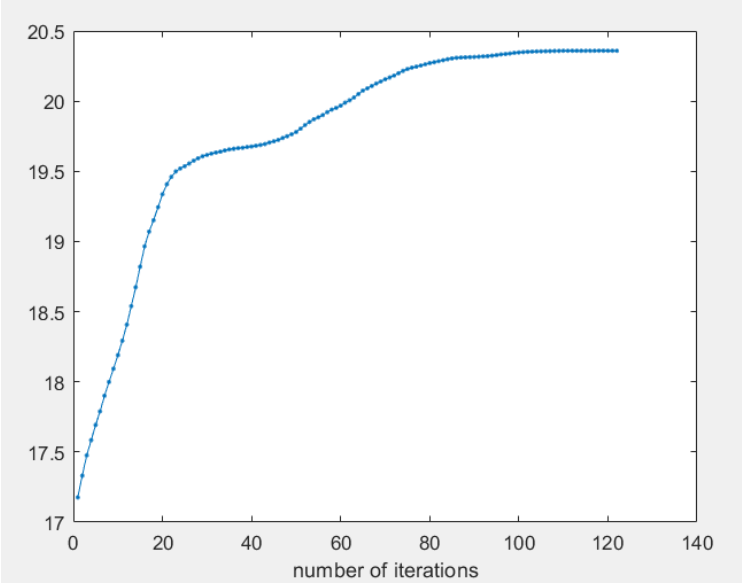
Παρατηρούμε ότι με αύξηση του αριθμού των bits ελαττώνεται η παραμόρφωση που παρατηρούμε και αύξηση στη τιμή του SQNR.

Οι γραφικές παραστάσεις της μεταβολής του SQNR σε σχέση με τον αριθμό των επαναλήψεων του αλγορίθμου Lloyd Max για την $AR_1(1)$ για N=2,4 και 8 bits αντίστοιχα είναι οι ακόλουθες:

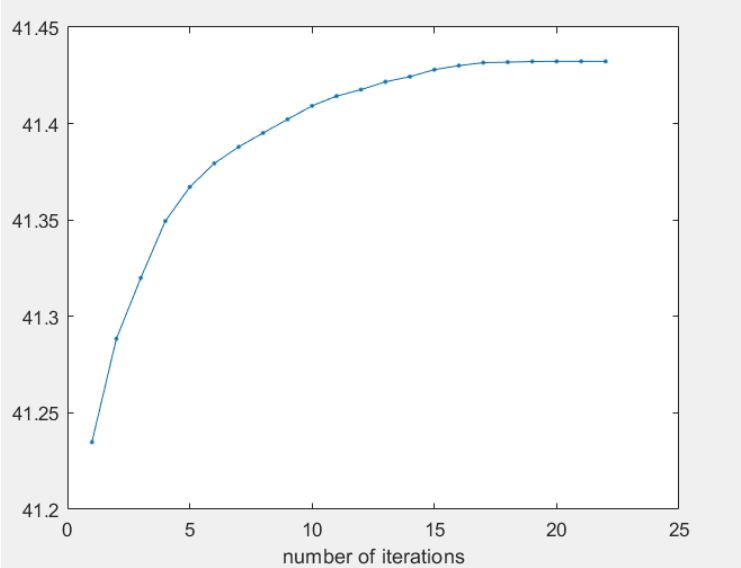
N=2 bits:



N=4 bits:

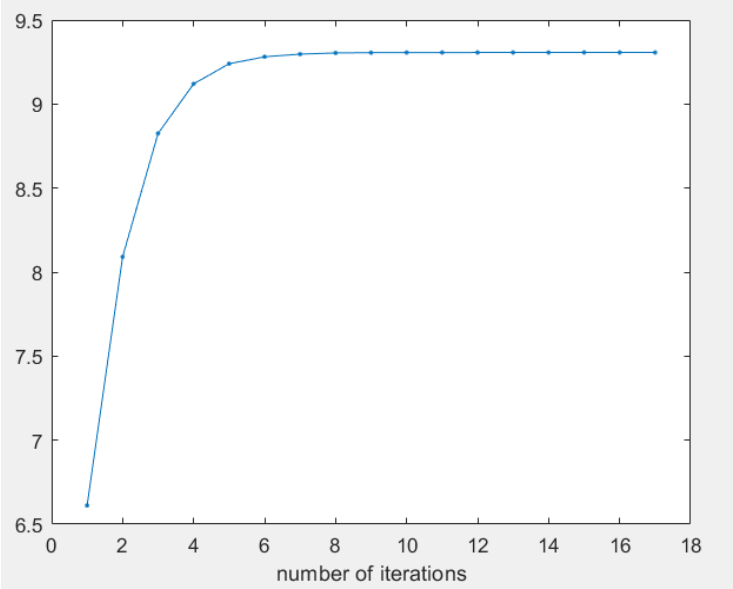


N=8 bits:

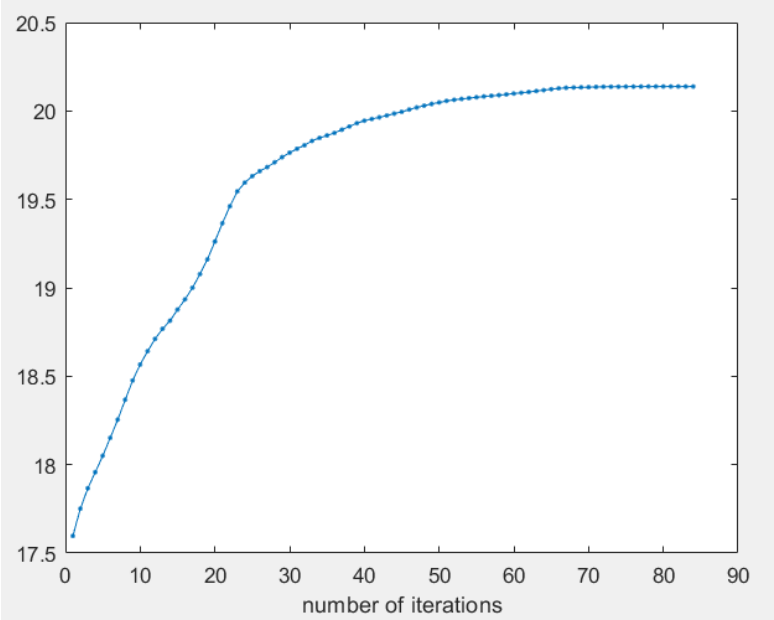


Οι γραφικές παραστάσεις της μεταβολής του SQNR σε σχέση με τον αριθμό των επαναλήψεων του αλγορίθμου Lloyd Max για την $AR_2(1)$ για N=2,4 και 8 bits αντίστοιχα είναι οι ακόλουθες:

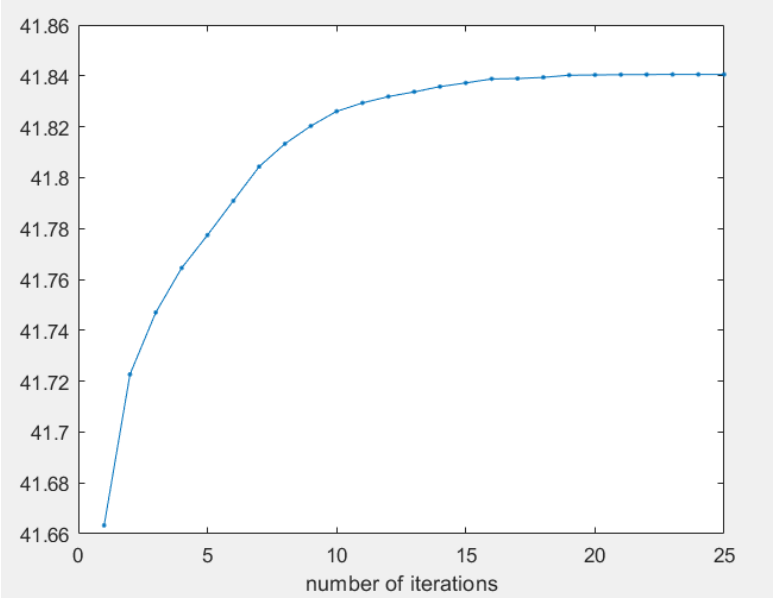
N=2 bits:



N=4 bits:



N=8 bits:



Οι τιμές του SQNR σε dBs που προέκυψαν χρησιμοποιώντας ADM είναι οι ακόλουθες:

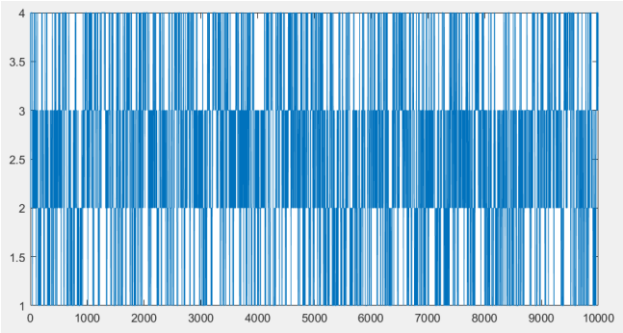
Διαδικασία	$\boldsymbol{SQNR}$
$AR_1(1)$	5.939558
$AR_2(1)$	0.065781

Παρατηρούμε ότι οι τιμές του SQNR που προέκυψαν χρησιμοποιώντας τον αλγόριθμο Lloyd Max για οποιοδήποτε

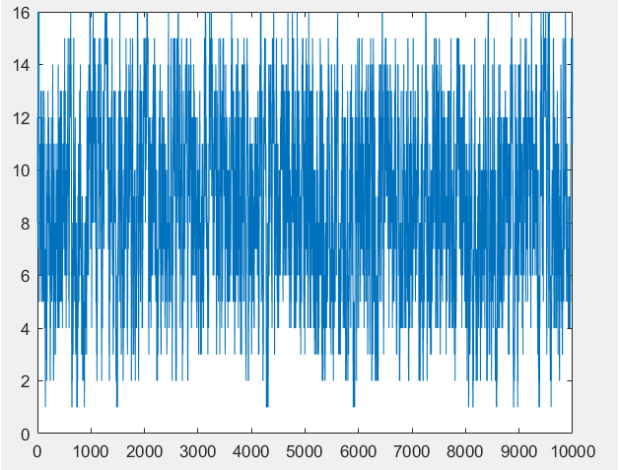
αριθμό bits και για τις δύο διαδικασίες είναι μεγαλύτερες σε σχέση με τις αντίστοιχες τιμές του ADM. Άρα, η μη ομοιόμορφη PCM παράγει καλύτερα αποτελέσματα από την ADM με τη λιγότερη δυνατή παραμόρφωση. Έχει όμως πιο περίπλοκη υλοποίηση και είναι πιο ακριβή.

1.1.β. Οι κυματομορφές εξόδου που προκύπτουν για τη διαδικασία $AR_1(1)$ με χρήση της μεθόδου PCM είναι οι ακόλουθες:

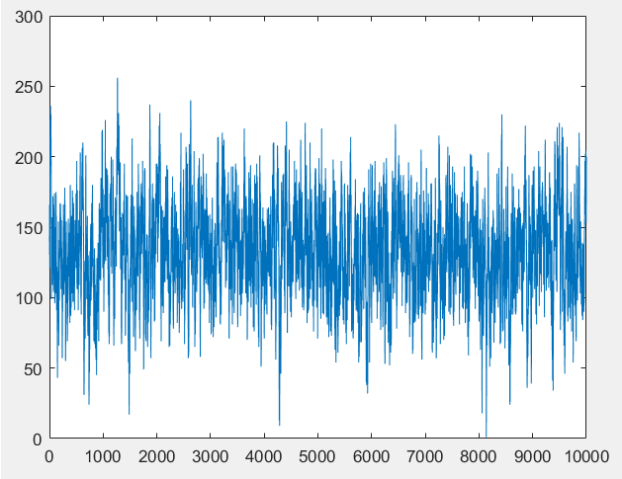
N=2 bits:



N=4 bits:

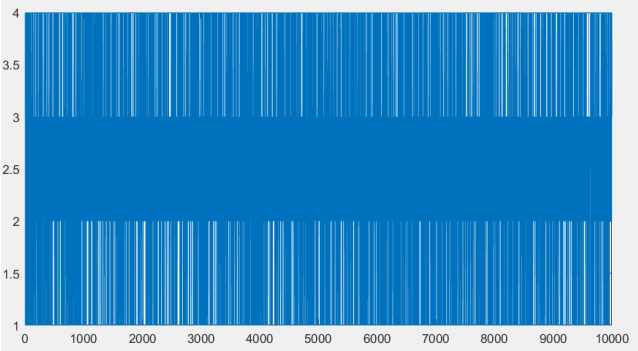


N=8 bits:

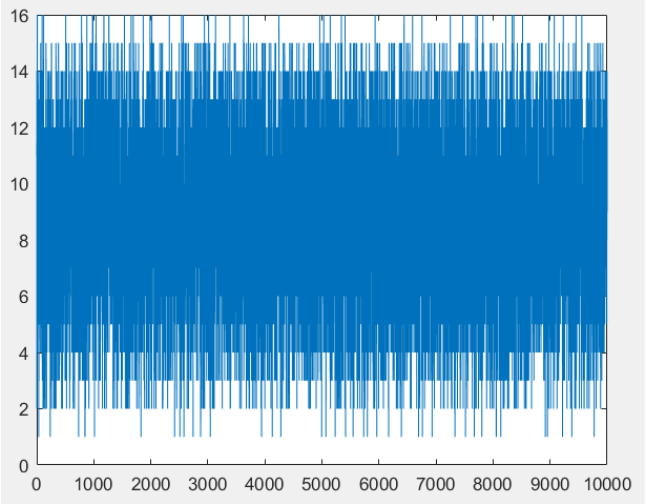


Οι κυματομορφές εξόδου που προκύπτουν για τη διαδικασία $AR_2(1)$ με χρήση της μεθόδου PCM είναι οι ακόλουθες:

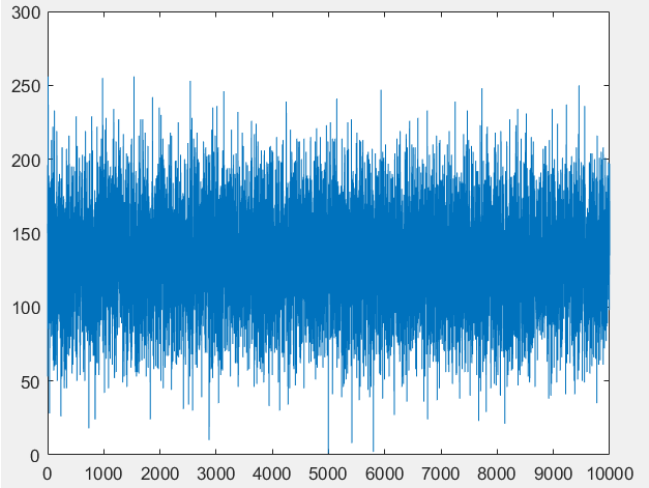
N=2 bits:



N=4 bits:

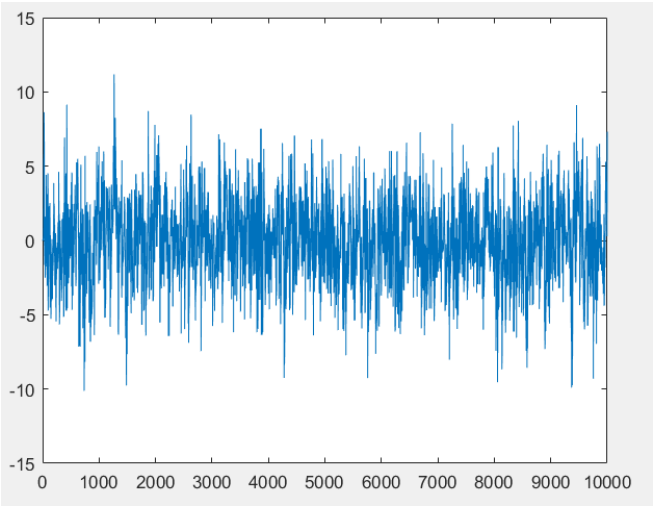


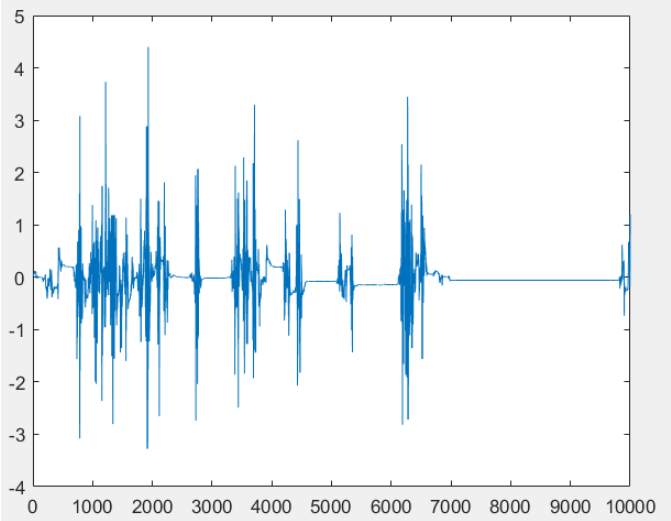
N=8 bits:



Παρατηρούμε με τη χρήση περισσότερων bits στη μέθοδο PCM παράγεται καλύτερη κυματομορφή εξόδου.

Οι κυματομορφές εξόδου που προκύπτουν για τις διαδικασίες $AR_1(1)$ και $AR_2(1)$ με χρήση της μεθόδου ADM είναι αντίστοιχα οι ακόλουθες:





Παρατηρούμε ότι στην περίπτωση της $AR_1(1)$ η κυματομορφή εξόδου που προκύπτει με τη μέθοδο ADM είναι πολύ καλύτερη σε σχέση με αυτές της μεθόδου PCM για N=2 και 4 bits, ενώ είναι αρκετά παρόμοια με αυτή που προκύπτει για N=8 bits. Από την άλλη, η κυματομορφή εξόδου της ADM με πηγή τη διαδικασία $AR_2(1)$ είναι αρκετά διαφορετική συγκριτικά με την αντίστοιχη κυματομορφή της μεθόδου PCM για N=8 bits.

1.2. Για την περίπτωση του PCM η εντροπία στην έξοδο του κβαντιστή για τη διαδικασία $AR_1(1)$ για N=2, 4 και 8 bits είναι η ακόλουθη:

N	**Εντροπία**
2	1.2985
4	3.1024
8	7.0725

Για την περίπτωση του PCM η εντροπία στην έξοδο του κβαντιστή για τη διαδικασία $AR_2(1)$ για N=2, 4 και 8 bits είναι η ακόλουθη:

N	**Εντροπία**
2	1.3500
4	3.1726
8	7.1427

1.3. Σύμφωνα με τα αποτελέσματα που βρέθηκαν στα δύο παραπάνω ερωτήματα, παρατηρούμε ότι η μέθοδος PCM παράγει μεγαλύτερο SQNR σε σχέση με την ADM ανεξάρτητα με το πόσα bits χρησιμοποιούνται, αν και αύξηση των bits προκαλεί και αύξηση του SQNR και συνεπώς καλύτερη ποιότητα σήματος εξόδου.

Επιπλέον, παρατηρήθηκε ότι τα αποτελέσματα με χρήση της μεθόδου ADM για τη διαδικασία $AR_2(1)$ ήταν πολύ χειρότερα και από τα αντίστοιχα του PCM, αλλά και από αυτά της διαδικασίας $AR_1(1)$ με ADM. Συγκεκριμένα και η τελική αναπαράσταση του σήματος εξόδου με πηγή την $AR_2(1)$ ήταν σε μικρότερο εύρος τιμών και δεν έμοιαζε καθόλου με το αρχικό, και οι τιμές του SQNR ήταν πολύ χαμηλότερες σε σχέση με όλες τις υπόλοιπες που μετρήθηκαν. Αντιθέτως, η μέθοδος PCM παρήγαγε πολύ καλά αποτελέσματα και για τις δύο διαδικασίες.

2.1.α. Για να βρούμε τη μέγιστη και την ελάχιστη τιμή της πηγής Β χρησιμοποιήθηκαν οι συναρτήσεις max και min της MatLab αντίστοιχα, από τις οποίες προέκυψαν οι εξής τιμές:

Οι τιμές του SQNR σε dBs που προέκυψαν μετά από K_{max} επαναλήψεις χρησιμοποιώντας PCM με N=2 και 4 bits είναι οι ακόλουθες: $min_{value} = -0.9453$ και $max_{value} = 0.9766$.

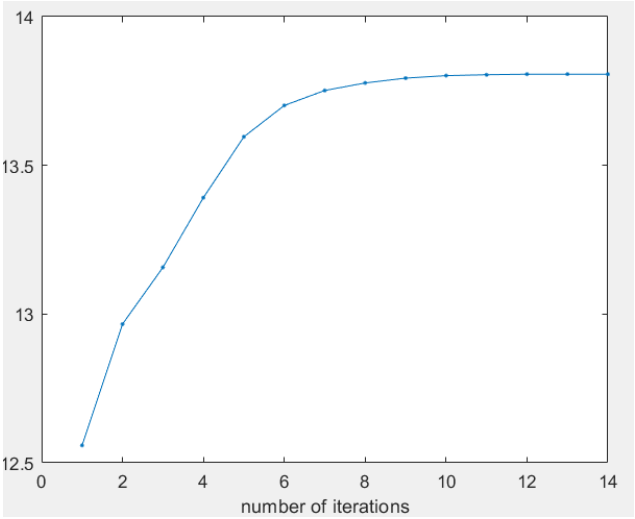
N	**SQNR**
2	13.806480
4	25.001167

Παρατηρούμε ότι με την αύξηση του αριθμού των bits στη μέθοδο PCM αυξάνεται και η τιμή του SQNR, όπως και ήταν αναμενόμενο.

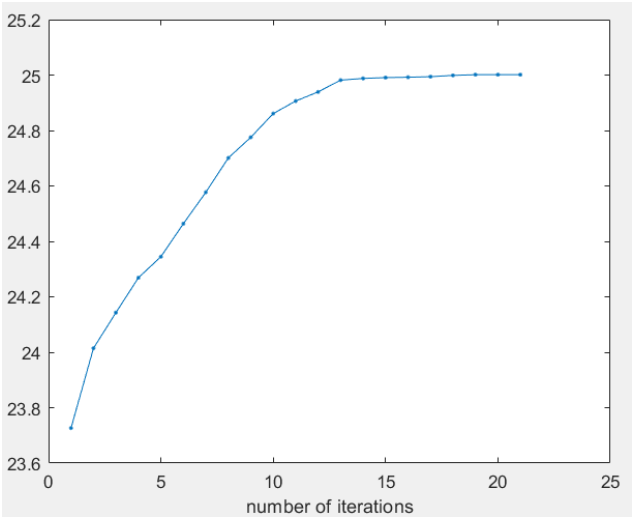
Οι γραφικές παραστάσεις της μεταβολής του SQNR σε σχέση με τον αριθμό των επαναλήψεων του αλγορίθμου Lloyd Max

για την πηγή B για N=2 και 4 bits αντίστοιχα είναι οι ακόλουθες:

N=2 bits:



N=4 bits:



Όσο λιγότερα είναι τα bits που έχουμε στην διάθεσή μας τόσο πιο μεγάλη είναι και η παραμόρφωση που παρατηρούμε. Αν αυξήσουμε τα bits που χρησιμοποιούμε, ο αλγόριθμος θα χρειαστεί περισσότερες επαναλήψεις μέχρι να τερματίσει και η τελική του τιμή θα έχει μεγαλύτερο SQNR.

2.1.β. Το τελικό οπτικό αποτέλεσμα αν μετατρέψουμε το κβαντισμένο διάνυσμα σε εικόνα διαστάσεων MxN, όπου

M=N=256 εκτελώντας y=reshape(x,M,N) για PCM με N=2 και 4 bits είναι το ακόλουθο:

N=2 bits:

N=4 bits:

Παρατηρούμε ότι με χρήση 4 bits, η τελική εικόνα που προκύπτει έχει πολύ καλύτερη ποιότητα και είναι πιο καθαρή, αφού έχει λιγότερο θόρυβο, γεγονός που επιβεβαιώνεται και από τις τιμές του SQNR.

2.2. Για την περίπτωση του PCM η εντροπία στην έξοδο του κβαντιστή για την πηγή Β για N=2 και 4 bits είναι η ακόλουθη:

N	**Εντροπία**
2	1.4992

4	3.1838

Παρατηρούμε ότι με την αύξηση των αριθμών των bits αυξάνεται και η εντροπία.

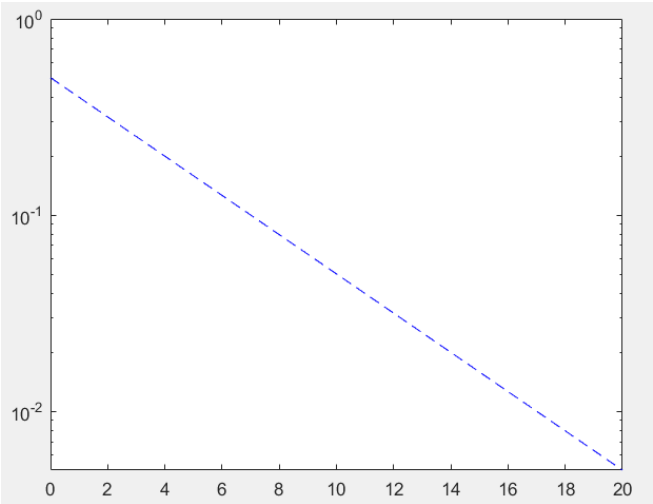
Ερωτήσεις-Ζητούμενα για Μέρος 2:

1. Η ακόλουθη περιγραφή του συστήματος M-PAM βασίζεται στην δοσμένη εκφώνηση. Ο κώδικας που βρίσκεται για το συγκεκριμένο ερώτημα στο παράρτημα, είναι για M=4 και 8, αφού στα επακόλουθα ερωτήματα αυτά είναι τα M που ζητούνται για απλή κωδικοποίηση.

Το σύστημα M-PAM δέχεται αρχικά ως είσοδο μία δυαδική ακολουθία. Αφού τη μετατρέψει σε σύμβολα με τη βοήθεια του mapper, την πολλαπλασιάζει με ορθογώνιο παλμό και τη μεταφέρει μέσω του διαμορφωτή. Έπειτα, προστίθεται λευκός θόρυβος Gaussian κατανομής μηδενικής μέσης τιμής και διασποράς $\sigma^2=\frac{N_0}{2}$ και αποδιαμορφώνεται. Το μονοδιάστατο διάνυσμα που προκύπτει εισέρχεται στον φωρατή, όπου μόλις αποφασιστεί πιο σύμβολο στάλθηκε, αντιστρέφεται μέσω του demapper σε bits.

2. Η πιθανότητα σφάλματος bit (Bit Error Rate-BER) είναι μία μέτρηση η οποία δείχνει τον αριθμό των λαθών που υπάρχουν στα bits που εκπέμπονται. Όσο μεγαλύτερη είναι, τόσο χειρότερη είναι η ροή των bits από το πομπό στο δέκτη και, επομένως, τόσο χειρότερη είναι η τελική λήψη των bits.

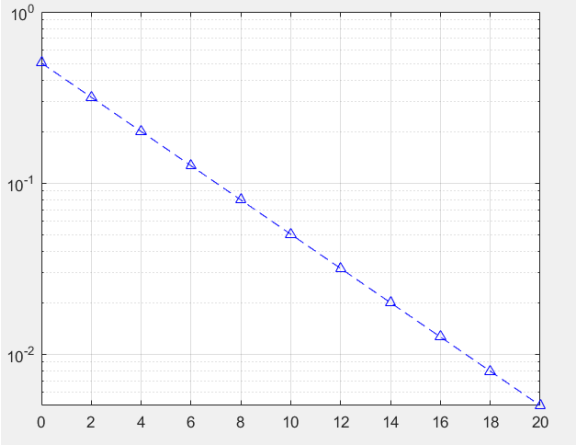
Η πιθανότητα σφάλματος bit και η καμπύλη BER που προκύπτουν από το σύστημα διαμόρφωσης M-PAM με M=4 και SNR=0:2:20dB είναι η ακόλουθη:



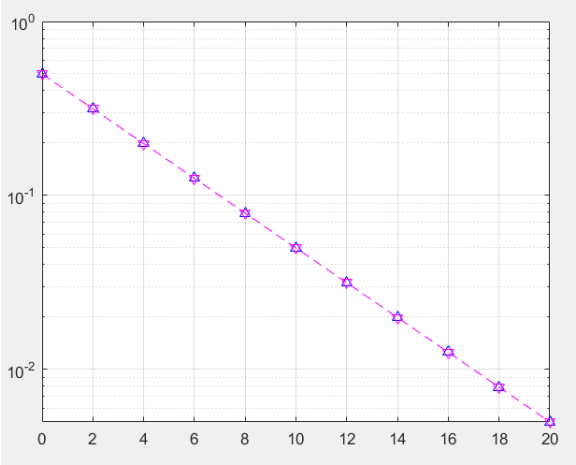
Όπου στον οριζόντιο άξονα είναι το SNR και στον κατακόρυφο είναι το BER.

3. Το σύστημα διαμόρφωσης M-PAM που δημιουργήθηκε είναι ζωνοπερατό και για τη μετάδοση των συμβόλων χρησιμοποιείται ορθογώνιος παλμός. Επομένως, το σύστημα μας ανήκει στην ειδική περίπτωση της μεταλλαγής ολίσθησης πλάτους (Amplitude Shift Keying- ASK). Η διαμόρφωση ASK χρησιμοποιεί ένα πεπερασμένο αριθμό από πλάτη, όπου το καθένα έχει ανατεθεί σε ένα μοναδικό συνδυασμό από δυαδικά ψηφία. Ανάλογα με την κυματομορφή εισόδου θα δημιουργηθεί και η κυματομορφή εξόδου με πλάτος που προκύπτει από την παραπάνω κωδικοποίηση. Έτσι, το παραγόμενο σήμα της ASK είναι εξαιρετικά ευαίσθητο σε εξωτερικούς παράγοντες, όπως είναι ο θόρυβος. Οπότε, έχει νόημα να χρησιμοποιηθεί κωδικοποίηση Gray.

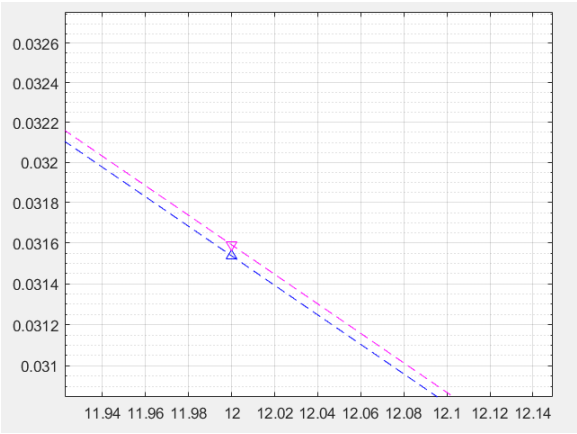
4. Η πιθανότητα σφάλματος bit και η καμπύλη BER που προκύπτουν από το σύστημα διαμόρφωσης M-PAM με M=8 για κωδικοποίηση κατά Gray και SNR=0:2:20dB είναι η ακόλουθη:



Αν στο ίδιο γράφημα προσθέσουμε και την καμπύλη BER που προκύπτει με M=8 για απλή κωδικοποίηση, λαμβάνουμε το ακόλουθο γράφημα:



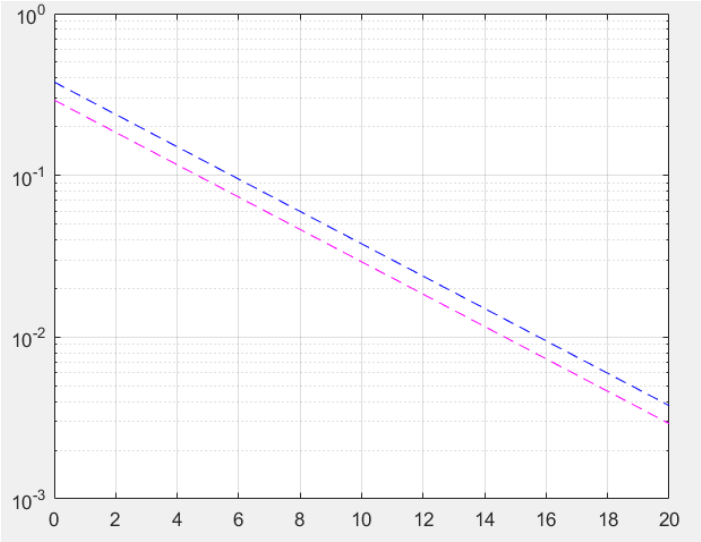
Επειδή η διαφορά δεν φαίνεται ξεκάθαρα, παρατίθεται και ένα zoomed-in γράφημα:



Όπου με μωβ είναι για M=8 και απλή κωδικοποίηση και μπλε M=8 και για κωδικοποίηση κατά Gray. Επίσης, στον οριζόντιο άξονα είναι το SNR και στον κατακόρυφο είναι το BER.

5. Η πιθανότητα σφάλματος συμβόλου (Symbol Error Rate-SER) είναι μία μέτρηση η οποία δείχνει την πιθανότητα εμφάνισης σφάλματος σε σύμβολο.

Η πιθανότητα σφάλματος συμβόλου και η καμπύλη SER που προκύπτουν από το σύστημα διαμόρφωσης M-PAM με M=4 και 8 για απλή κωδικοποίηση και SNR=0:2:20dB είναι η ακόλουθη:



Όπου με μωβ είναι για M=8 και μπλε για M=4. Επίσης, στον οριζόντιο άξονα είναι το SNR και στον κατακόρυφο είναι το SER.

Παράρτημα για τους κώδικες:

Μέρος 1:

Lloyd-Max:

```
function [xq, centers, D] = lloyd_max(x, N, minv, maxv)

s = size(x);
s = s(1);
l=2^N;
```

```
d=abs(maxv-minv)/l;

centers(1)=maxv - d/2;
for i=2:l
centers(i)=centers(i-1)-d;
end

if min(x) < minv
    for i=1:s
        if x(i) < minv
            x(i) = minv;
        end
    end
end
if max(x) > maxv
    for i=1:s
        if x(i) > maxv
            x(i) = maxv;
        end
    end
end

centers = flip(centers);
centers = [minv centers maxv];

D = [0 1];

k = 2;
while abs(D(k) - D(k-1)) >= eps
    xq = [];
    total = 0;
    steps   = zeros(length(centers));
    range = zeros(length(centers));

    T = [];
    T(1) = minv;
    for i=2:(length(centers)-2)
        T(i) = (centers(i) + centers(i+1))/2;
    end
    T(i+1) = maxv;

    for i=1:s
        for j=1:(length(T)-1)
            if T(j) < x(i) && x(i) <= T(j+1)
                xq(i) = j;
                total = total + abs(centers(j+1) - x(i));
                range(j) = range(j) + x(i);
                steps(j)   = steps(j) + 1;
            end
        end
        if x(i) == T(1)
            xq(i) = 1;
            total = total + abs(centers(2) - x(i));
            range(1) = range(1) + x(i);
            steps(1)   = steps(1) + 1;
        end
    end
    averaged = total/s;
```

```
    D = [D averaged];
    k = k + 1;

    for j=2:(length(centers)-1)
        if steps(j-1) ~= 0
            centers(j) = range(j-1)/steps(j-1);
        end
    end
end

D(1) = [];
D(2) = [];

centers(1) = [];
centers(length(centers)) = [];

xq = xq';
fprintf('Successfully exited after %d iterations.\n', k-2);
```

ADM:

```
function [iq, sqnr] = adm(input,con,initialstep)

l = size(input);
y = [];
iq = [];
out = [];
delta = [];
k=0;
delta(1) = initialstep;
if input(1) >= 0
    out(1) = 1;
else
    out(1) = -1;
end
y(1) = out(1) * delta(1);
iq(1) = y(1);
for n = 2:1:l
    if input(n) >= iq(n-1)
        out(n) = 1;
    else
        out(n) = -1;
    end
    delta(n) = delta(n-1) * con ^ (out(n)*out(n-1));
    y(n) = out(n) * delta(n);
    iq(n) = iq(n-1) + y(n);
    k=k+1;
end
iq=iq';
sqnr = 10*log10(mean(input.^2)/mean((input-iq).^2));
fprintf('Successfully exited after %d iterations.\n', k);
```

$AR_1(1)$ και $AR_2(1)$:

```
L=10000;
```

```
x=randn(L,1);
a1=0.9;
c1=0.01;
b=1;
a=[1 -a1]';
c=[1 -c1]';
y1=filter(b,a,x);
y2=filter(b,c,x);
```

Υπολογισμός του SQNR για το PCM για $AR_1(1)$ και $AR_2(1)$:

```
%%AR1(1) N=2:
[xq, centers, D] = lloyd_max(y1, 2, min(y1), max(y1));
new = centers(xq);
new = new';
a = mean(y1.^2);
b = mean((new-y1).^2);
SNR = 10 * log10(a/b);

fprintf('SQNR of 2-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

%%AR1(1) N=4:
[xq, centers, D] = lloyd_max(y1, 4, min(y1), max(y1));
new = centers(xq);
new = new';
a = mean(y1.^2);
b = mean((new-y1).^2);
SNR = 10 * log10(a/b);

fprintf('SQNR of 4-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

%%AR1(1) N=8:
[xq, centers, D] = lloyd_max(y1, 8, min(y1), max(y1));
new = centers(xq);
new = new';
a = mean(y1.^2);
b = mean((new-y1).^2);
SNR = 10 * log10(a/b);

fprintf('SQNR of 8-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

%%AR2(1) N=2:
[xq, centers, D] = lloyd_max(y2, 2, min(y2), max(y2));
new = centers(xq);
new = new';
a = mean(y2.^2);
b = mean((new-y2).^2);
SNR = 10 * log10(a/b);

fprintf('SQNR of 2-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

%%AR2(1) N=4:
[xq, centers, D] = lloyd_max(y2, 4, min(y2), max(y2));
new = centers(xq);
new = new';
a = mean(y2.^2);
b = mean((new-y2).^2);
SNR = 10 * log10(a/b);
```

```
fprintf('SQNR of 4-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

%%AR2(1) N=8:
[xq, centers, D] = lloyd_max(y2, 8, min(y2), max(y2));
new = centers(xq);
new = new';
a = mean(y2.^2);
b = mean((new-y2).^2);
SNR = 10 * log10(a/b);

fprintf('SQNR of 8-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);
```

Υπολογισμός του SQNR για την ADM για $AR_1(1)$ και $AR_2(1)$:

```
%%AR1(1):
[xq, sqnr] = adm(y1, 1.5, 0.01);
plot(xq);
fprintf('SQNR of quantization (using ADM) is %f dBs\n', sqnr);

%%AR2(1):
[xq, sqnr] = adm(y2, 1.5, 0.01);
plot(xq);

fprintf('SQNR of quantization (using ADM) is %f dBs\n', sqnr);
```

Υπολογισμός της εντροπίας για το PCM για $AR_1(1)$ και $AR_2(1)$:

```
%%AR1(1) N=2:
[xq, centers, D] = lloyd_max(y1, 2, min(y1), max(y1));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
entropyPCM = - sum(pr .* log2(pr));
fprintf('The entropy with 2-bit is: %d\n', entropyPCM);
%%AR1(1) N=4:
[xq, centers, D] = lloyd_max(y1, 4, min(y1), max(y1));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
entropyPCM = - sum(pr .* log2(pr));
fprintf('The entropy with 4-bit is: %d\n', entropyPCM);
%%AR1(1) N=8:
[xq, centers, D] = lloyd_max(y1, 8, min(y1), max(y1));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
mp = pr(pr>0);
entropyPCM = - sum(mp .* log2(mp));
fprintf('The entropy with 8-bit is: %d\n', entropyPCM);
%%AR2(1) N=2:
[xq, centers, D] = lloyd_max(y2, 2, min(y2), max(y2));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
entropyPCM = - sum(pr .* log2(pr));
```

```
fprintf('The entropy with 2-bit is: %d\n', entropyPCM);
%%AR2(1) N=4:
[xq, centers, D] = lloyd_max(y2, 4, min(y2), max(y2));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
entropyPCM = - sum(pr .* log2(pr));
fprintf('The entropy with 4-bit is: %d\n', entropyPCM);
%%AR2(1) N=8:
[xq, centers, D] = lloyd_max(y2, 8, min(y2), max(y2));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
mp = pr(pr>0);
entropyPCM = - sum(mp .* log2(mp));
fprintf('The entropy with 8-bit is: %d\n', entropyPCM);
```

Υπολογισμός του SQNR για την πηγή B:

```
z = i(:);
z = (z-128)/128;
[xq, centers, D] = lloyd_max(z, 2, min(z), max(z));
new = centers(xq);
new = new';
a = mean(z.^2);
b = mean((new-z).^2);
SNR = 10 * log10(a/b);
xq = 128*xq+128;
z1=reshape(xq,256,256);
imshow(z1, []);


fprintf('SQNR of 2-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);

[xq, centers, D] = lloyd_max(z, 4, min(z), max(z));
new = centers(xq);
new = new';
a = mean(z.^2);
b = mean((new-z).^2);
SNR = 10 * log10(a/b);
z1=reshape(xq,256,256);
imshow(z1, []);
fprintf('SQNR of 4-bit quantization (using Lloyd-Max) is %f dBs\n', SNR);
```

Υπολογισμός της εντροπίας για την πηγή B:

```
%%N=2:
[xq, centers, D] = lloyd_max(z, 2, min(z), max(z));
v = tabulate(xq);
pr = v(:,3);
pr = pr ./ 100;
entropypcm = - sum(pr .* log2(pr));
fprintf('The entropy with 2-bit is: %d\n', entropypcm);


%%N=4:
[xq, centers, D] = lloyd_max(z, 4, min(z), max(z));
v = tabulate(xq);
```

```
pr = v(:,3);
pr = pr ./ 100;
entropypcm = - sum(pr .* log2(pr));
fprintf('The entropy with 4-bit is: %d\n', entropypcm);
```

Μέρος 2:

Mapper:

```
function input_symbols=mapper(seq,M)

if M==4
    l=floor((length(seq)/2));
elseif M==8
    l=floor((length(seq)/3));
end

input_symbols=zeros(l,1);
i=1;
if M==4
    for j=1:2:length(seq)
    input_symbols(i)=2*seq(j)+seq(j+1);
    i=i+1;
    end
elseif M==8
    for j=1:3:length(seq)-1
    input_symbols(i)=4*seq(j)+2*seq(j+1)+seq(j+2);
    i=i+1;
    end
end


end
```

Διαμορφωτής:

```
function diam=modulator(symbols,M)

Tc=4;
fc=0.25;
Ts=40;
Es=1;

gt=sqrt((2*Es)/Ts);
A=sqrt(3/(M^2-1));
Am=(2*symbols - (M+1))*A;

diam=Am*gt*cos(2*pi*fc*(1:Ts));
end
```

Κανάλι AWGN:

```
function output_s = AWGN(input_s,SNR,M)
[Lb,Ts]=size(input_s);
v=1/(2*log2(M)*(10^(SNR/10)));
noise_awgn=sqrt(v)*randn(Lb,Ts);
output_s=input_s+noise_awgn;
```

```
end
```

Αποδιαμορφωτής:

```
function apod=demodulator(diam,M)

Tc=4;
fc=0.25;
Ts=40;
Es=1;

gt=sqrt((2*Es)/Ts);
A=sqrt((M^2-1)/3);
Am=((diam+(M+1))/2)*A;

apod=Am*gt*cos(2*pi*fc*(1:Ts))';
end
```

Φωρατής:

```
function fwr=detector(r,M)
A=sqrt(3/(M^2-1));

if M==4
    s=[-3,-1,1,3].*A;
    fwr=zeros(length(r),2);
for i=1:length(r)
    if(r(i) == -3/sqrt(5))
        fwr(i,1) = 0;
        fwr(i,2) = 0;
    elseif(r(i) == -1/sqrt(5))
        fwr(i,1) = 0;
        fwr(i,2) = 1;
    elseif(r(i) == 1/sqrt(5))
        fwr(i,1) = 1;
        fwr(i,2) = 0;
    elseif(r(i) == 3/sqrt(5))
        fwr(i,1) = 1;
        fwr(i,2) = 1;
    end
end
elseif M==8
    s=[-7,-5,-3,-1,1,3,5,7].*A;

fwr=zeros(length(r),3);
for i=1:length(r)
    if(r(i) == -7/sqrt(3/63))
        fwr(i,1) = 0;
        fwr(i,2) = 0;
        fwr(i,3) = 0;
    elseif(r(i) == -5/sqrt(3/63))
        fwr(i,1) = 0;
        fwr(i,2) = 0;
        fwr(i,3) = 1;
    elseif(r(i) == -3/sqrt(3/63))
        fwr(i,1) = 0;
        fwr(i,2) = 1;
        fwr(i,3) = 0;
    elseif(r(i) == -1/sqrt(3/63))
```

```
        fwr(i,1) = 0;
        fwr(i,2) = 1;
        fwr(i,3) = 1;
    elseif(r(i) == 1/sqrt(3/63))
        fwr(i,1) = 1;
        fwr(i,2) = 0;
        fwr(i,3) = 0;
    elseif(r(i) == 3/sqrt(3/63))
        fwr(i,1) = 1;
        fwr(i,2) = 0;
        fwr(i,3) = 1;
    elseif(r(i) == 5/sqrt(3/63))
        fwr(i,1) = 1;
        fwr(i,2) = 1;
        fwr(i,3) = 0;
    elseif(r(i) == 7/sqrt(3/63))
        fwr(i,1) = 1;
        fwr(i,2) = 1;
        fwr(i,3) = 1;
    end
end
end

end
```

Demapper:

```
function output_seq=demapper(symbols,M)

if M==4
    l=2*length(symbols);
elseif M==8
    l=3*length(symbols);
end

output_seq=zeros(l,1);
i=1;

if M==4
    for j=1:2:l
    output_seq(j)=floor(symbols(i)/2);
    output_seq(j+1)=rem(symbols(i),2);
    i=i+1;
    end
elseif M==8
    for j=1:3:l
    output_seq(j)=ceil(rem(symbols(i),2));
    output_seq(j+1)=floor((rem(symbols(i),4))/2);
    output_seq(j+1)=ceil(symbols(i)/8);
    i=i+1;
    end
end


end
```

Υπολογισμός BER για M=4 για απλή κωδικοποίηση:

```
Lb=10^5;
```

```
SNR=(0:2:20);
sl=length(SNR);
BER_temp=zeros(sl);
BER=zeros(sl);
input_s=randsrc(Lb,1,[0 1]);
input_symbols=mapper(input_s,4);
diam=modulator(input_symbols,4);

for i=1:sl
    output_s=AWGN(diam,SNR(i),4);
    apod=demodulator(output_s,4);
    fwr=detector(apod,4);
    output_seq=demapper(fwr,4);
    BER_temp(i)=biterr(input_s,output_seq)/Lb;
    BER(i)=BER_temp(i)*(1/(10^(SNR(i)/10)));
end

pl=zeros(sl,1);
for k=1:sl
    pl(k)=10*log10(10^(SNR(k)/10));
end

semilogy(pl,BER,'b--');
```

Mapper για κωδικοποίηση κατά Gray:

```
function input_symbols=gray_mapper(seq)
%gia thn gray kwdikopoihsh exoume mono M=8

l=floor((length(seq)/3));

input_symbols=zeros(l,1);
i=1;

for j=1:3:l
    input_symbols(i)=seq(j)+2*seq(j+1)+4*seq(j+2);
    i=i+1;
end

input_symbols=bin2gray(input_symbols,'pam',8);

end
```

Demapper για κωδικοποίηση κατά Gray:

```
function output_seq=gray_demapper(symbols)
%gia thn gray kwdikopoihsh exoume mono M=8

l=3*length(symbols);
symbols=gray2bin(symbols,'pam',8);

output_seq=zeros(l,1);
i=1;

for j=1:3:l
    output_seq(j)=ceil(rem(symbols(i),2));
    output_seq(j+1)=floor((rem(symbols(i),4))/2);
    output_seq(j+1)=ceil(symbols(i)/8);
    i=i+1;
```

```
end


end
```

Υπολογισμός BER για M=8 για απλή κωδικοποίηση και κωδικοποίηση κατά Gray:

```
Lb=10^5-1;
SNR=(0:2:20);
sl=length(SNR);
BER_temp=zeros(sl);
BER=zeros(sl);
input_s=randsrc(Lb,1,[0 1]);
input_symbols=gray_mapper(input_s);
diam=modulator(input_symbols,8);

for i=1:sl
    output_s=AWGN(diam,SNR(i),8);
    apod=demodulator(output_s,8);
    fwr=detector(apod,8);
    output_seq=gray_demapper(fwr);
    BER_temp(i)=biterr(input_s,output_seq)/Lb;
    BER(i)=BER_temp(i)*(1/(10^(SNR(i)/10)));
end

BER_temps=zeros(sl);
BERs=zeros(sl);
input_ss=randsrc(Lb,1,[0 1]);
input_symbolss=mapper(input_ss,8);
diams=modulator(input_symbolss,8);

for i=1:sl
    output_ss=AWGN(diams,SNR(i),8);
    apods=demodulator(output_ss,8);
    fwrs=detector(apods,8);
    output_seqs=demapper(fwrs,8);
    BER_temps(i)=biterr(input_ss,output_seqs)/Lb;
    BERs(i)=BER_temps(i)*(1/(10^(SNR(i)/10)));
end

pl=zeros(sl,1);
for k=1:sl
    pl(k)=10*log10(10^(SNR(k)/10));
end

figure
semilogy(pl,BER,'b^--');
hold on;
semilogy(pl,BERs,'mv--');
hold off;
grid on;
```

Υπολογισμός SER για M=4 και 8 για απλή κωδικοποίηση:

```
Lb=10^5;
Lbs=10^5-1;
SNR=(0:2:20);
```

```
sl=length(SNR);
SER_temp=zeros(sl);
SER=zeros(sl);
input_s=randsrc(Lb,1,[0 1]);
input_symbols=mapper(input_s,4);
diam=modulator(input_symbols,4);

for i=1:sl
    output_s=AWGN(diam,SNR(i),4);
    apod=demodulator(output_s,4);
    fwr=detector(apod,4);
    output_seq=demapper(fwr,4);
    SER_temp(i)=symerr(input_symbols,fwr(:,1))/Lb;
    SER(i)=SER_temp(i)*(1/(10^(SNR(i)/10)));
end

SER_temps=zeros(sl);
SERs=zeros(sl);
input_ss=randsrc(Lbs,1,[0 1]);
input_symbolss=mapper(input_ss,8);
diams=modulator(input_symbolss,8);

for i=1:sl
    output_ss=AWGN(diams,SNR(i),8);
    apods=demodulator(output_ss,8);
    fwrs=detector(apods,8);
    output_seqs=demapper(fwrs,8);
    SER_temps(i)=symerr(input_symbolss,fwrs(:,1))/Lbs;
    SERs(i)=SER_temps(i)*(1/(10^(SNR(i)/10)));
end

pl=zeros(sl,1);
for k=1:sl
    pl(k)=10*log10(10^(SNR(k)/10));
end

figure
semilogy(pl,SER,'b--');
hold on;
semilogy(pl,SERs,'m--');
hold off;
grid on;
```